Terms used **time** and **frequency** **near5** **garbage collection** and **verbosegc**

Found **241,249** of **1,001,690**

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The Digital Library

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Design and evaluation of dynamic optimizations for a Java just-in-time compiler

Toshio Suganuma, Toshiaki Yasue, Motohiro Kawahito, Hideaki Komatsu, Toshio Nakatani

July 2005 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 27 Issue 4

Publisher: ACM Press

Full text available: pdf(1.60 MB) Additional Information: full citation, abstract, references, citings, index terms

The high performance implementation of Java Virtual Machines (JVM) and Just-In-Time (JIT) compilers is directed toward employing a dynamic compilation system on the basis of online runtime profile information. The trade-off between the compilation overhead and performance benefit is a crucial issue for such a system. This article describes the design and implementation of a dynamic optimization framework in a production-level Java JIT compiler, together with two techniques for profile-directed o ...

Keywords: JIT compiler, Recompilation, adaptive optimization, code specialization, dynamic compilation, profile-directed method inlining

2 Controlling garbage collection and heap growth to reduce the execution time of Java applications

Tim Brecht, Eshrat Arjomandi, Chang Li, Hang Pham

September 2006 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 28 Issue 5

Publisher: ACM Press

Full text available: pdf(335.24 KB) Additional Information: full citation, abstract, references, index terms

In systems that support garbage collection, a tension exists between collecting garbage too frequently and not collecting it frequently enough. Garbage collection that occurs too frequently may introduce unnecessary overheads at the risk of not collecting much garbage during each cycle. On the other hand, collecting garbage too infrequently can result in applications that execute with a large amount of virtual memory (i.e., with a large footprint) and suffer from increased execution times due to ...

Keywords: Garbage collection, Java, heap growth, implementation, memory management, performance measurement, programming languages

3 A region-based compilation technique for dynamic compilers

Toshio Suganuma, Toshiaki Yasue, Toshio Nakatani

January 2006 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 28 Issue 1

Publisher: ACM Press

Full text available: pdf(977.63 KB) Additional Information: full citation, abstract, references, index terms

Method inlining and data flow analysis are two major optimization components for effective program transformations, but they often suffer from the existence of rarely or never executed code contained in the target method. One major problem lies in the assumption that the compilation unit is partitioned at method boundaries. This article describes the design and implementation of a region-based compilation technique in our dynamic optimization framework, in which the compiled regions are selected ...

Keywords: JIT compiler, Region-based compilation, dynamic compilation, on-stack replacement, partial inlining

4 Eventrons: a safe programming construct for high-frequency hard real-time applications

Daniel Spoonhower, Joshua Auerbach, David F. Bacon, Perry Cheng, David Grove

June 2006 **ACM SIGPLAN Notices , Proceedings of the 2006 ACM SIGPLAN conference on Programming language design and implementation PLDI '06**, Volume 41 Issue 6

Publisher: ACM Press

Full text available: pdf(339.16 KB) Additional Information: full citation, abstract, references, citings, index terms

While real-time garbage collection has achieved worst-case latencies on the order of a millisecond, this technology is approaching its practical limits. For tasks requiring extremely low latency, and especially periodic tasks with frequencies above 1 KHz, Java programmers must currently resort to the NoHeapRealtimeThread construct of the Real-Time Specification for Java. This technique requires expensive run-time checks, can result in unpredictable low-level exceptions, and inhibits communicatio ...

Keywords: allocation, real-time, scheduling

5 Time-triggered garbage collection: robust and adaptive real-time GC scheduling for embedded systems

Sven Gestegard Robertz, Roger Henriksson

June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems LCTES '03**, Volume 38 Issue 7

Publisher: ACM Press

Full text available: pdf(325.89 KB) Additional Information: full citation, abstract, references, citings, index terms

The advent of Java and similar languages on the real-time system scene necessitates the development of efficient strategies for scheduling the work of a garbage collector in a non-intrusive way. We propose a scheduling strategy, *time-triggered garbage collection*, based on assigning the collector a deadline for when it must complete its current cycle.We show that a time-triggered GC with fixed deadline can have equal or better real-time performance than an allocation-triggered GC, which is ...

Keywords: auto-tuning, embedded systems, garbage collection, real-time, scheduling

6 Creating and preserving locality of java applications at allocation and garbage collection times

Yefim Shuf, Manish Gupta, Hubertus Franke, Andrew Appel, Jaswinder Pal Singh

November 2002 **ACM SIGPLAN Notices , Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '02**, Volume 37 Issue 11

Publisher: ACM Press

Full text available: pdf(180.20 KB) Additional Information: full citation, abstract, references, citings, index terms

The growing gap between processor and memory speeds is motivating the need for optimization strategies that improve data locality. A major challenge is to devise techniques suitable for pointer-intensive applications. This paper presents two techniques aimed at improving the memory behavior of pointer-intensive applications with dynamic memory allocation, such as those written in Java. First, we present an allocation time object placement technique based on the recently introduced notion of *p* ...

Keywords: *JVM, Java, garbage collection, heap traversal, locality, locality based graph traversal, memory allocation, memory management, object co-allocation, object placement, prolific types, run-time systems*

7 Myths and realities: the performance impact of garbage collection

Stephen M. Blackburn, Perry Cheng, Kathryn S. McKinley

June 2004 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the joint international conference on Measurement and modeling of computer systems SIGMETRICS '04/Performance '04**, Volume 32 Issue 1

Publisher: ACM Press

Full text available: pdf(305.06 KB) Additional Information: full citation, abstract, references, citings, index terms, review

This paper explores and quantifies garbage collection behavior for three whole heap collectors and generational counterparts: *copying semi-space, mark-sweep,* and *reference counting*, the canonical algorithms from which essentially all other collection algorithms are derived. Efficient implementations in MMTk, a Java memory management toolkit, in IBM's Jikes RVM share all common mechanisms to provide a clean experimental platform. Instrumentation separates collector and program behav ...

Keywords: generational, java, mark-sweep, reference counting, semi-space

8 Tuning garbage collection for reducing memory system energy in an embedded java environment

G. Chen, R. Shetty, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, M. Wolczko

November 2002 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 1 Issue 1

Publisher: ACM Press

Full text available: pdf(740.23 KB) Additional Information: full citation, abstract, references, citings, index terms

Java has been widely adopted as one of the software platforms for the seamless integration of diverse computing devices. Over the last year, there has been great momentum in adopting Java technology in devices such as cellphones, PDAs, and pagers where optimizing energy consumption is critical. Since, traditionally, the Java virtual machine (JVM), the cornerstone of Java technology, is tuned for performance, taking into account energy consumption requires reevaluation, and possibly redesign of t ...

Keywords: Garbage collector, Java Virtual Machine (JVM), K Virtual Machine (KVM), low

power computing

9 MC²: high-performance garbage collection for memory-constrained environments

Narendran Sachindran, J. Eliot B. Moss, Emery D. Berger
October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '04**, Volume 39 Issue 10
Publisher: ACM Press

Full text available: pdf(503.53 KB) Additional Information: full citation, abstract, references, citings, index terms

Java is becoming an important platform for memory-constrained consumer devices such as PDAs and cellular phones, because it provides safety and portability. Since Java uses garbage collection, efficient garbage collectors that run in constrained memory are essential. Typical collection techniques used on these devices are mark-sweep and mark-compact. Mark-sweep collectors can provide good throughput and pause times but suffer from fragmentation. Mark-compact collectors prevent fragmentation, ...

Keywords: copying collector, generational collector, java, mark-compact, mark-copy, mark-sweep, memory-constrained copying

10 A unified theory of garbage collection

David F. Bacon, Perry Cheng, V. T. Rajan
October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '04**, Volume 39 Issue 10
Publisher: ACM Press

Full text available: pdf(223.52 KB) Additional Information: full citation, abstract, references, index terms

Tracing and reference counting are uniformly viewed as being fundamentally different approaches to garbage collection that possess very distinct performance properties. We have implemented high-performance collectors of both types, and in the process observed that the more we optimized them, the more similarly they behaved - that they seem to share some deep structure.

We present a formulation of the two algorithms that shows that they are in fact duals of each other. Intuitively, the ...

Keywords: graph algorithms, mark-and-sweep, reference counting, tracing

11 Syncopation: generational real-time garbage collection in the metronome

David F. Bacon, Perry Cheng, David Grove, Martin T. Vechev
June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems LCTES '05**, Volume 40 Issue 7
Publisher: ACM Press

Full text available: pdf(212.34 KB) Additional Information: full citation, abstract, references, citings, index terms

Real-time garbage collection has been shown to be feasible, but for programs with high allocation rates, the utilization achievable is not sufficient for some systems.Since a high allocation rate is often correlated with a more high-level, abstract programming style, the ability to provide good real-time performance for such programs will help continue to raise the level of abstraction at which real-time systems can be programmed.We have

developed techniques that allow generational collection to ...

Keywords: allocation, garbage collection, real-time, scheduling

12 Tree Rerooting in Distributed Garbage Collection: Implementation and Performance Evaluation

Luc Moreau

December 2001 **Higher-Order and Symbolic Computation**, Volume 14 Issue 4

Publisher: Kluwer Academic Publishers

Full text available: Publisher Site Additional Information: full citation, abstract, references, citings, index terms

We have recently defined a new algorithm for distributed garbage collection based on reference-counting (Luc Moreau, in *Proceedings of the Third International Conference of Functional Programming (ICFP'98)*, Sept. 1998, pp. 204-215; Luc Moreau and J. Duprat, Technical Report RR1999-18, Ecole Normale Supérieure, Lyon, March 1999). At the heart of the algorithm, we find *tree rerooting*, a mechanism able to reduce third-party dependencies by reorganising diffusion t ...

Keywords: benchmark, distributed garbage collection, distributed reference counting, performance evaluation

13 Cost-effective object space management for hardware-assisted real-time garbage collection

Kelvin D. Nilsen, William J. Schmidt

December 1992 **ACM Letters on Programming Languages and Systems (LOPLAS)**, Volume 1 Issue 4

Publisher: ACM Press

Full text available: pdf(1.29 MB) Additional Information: full citation, abstract, references, citings, index terms

Modern object-oriented languages and programming paradigms require finer-grain division of memory than is provided by traditional paging and segmentation systems. This paper describes the design of an OSM (Object Space Manager) that allows partitioning of real memory on object, rather than page, boundaries. The time required by the OSM to create an object, or to find the beginning of an object given a pointer to any location within it, is approximately one memory cycle. Object sizes are lim ...

Keywords: automatic garbage collection, dynamic storage management, high-level language architectures, memory technologies, real-time and embedded systems, run-time environments

14 New garbage collection algorithms and strategies: Dynamic selection of application-specific garbage collectors

Sunil Soman, Chandra Krintz, David F. Bacon

October 2004 **Proceedings of the 4th international symposium on Memory management ISMM '04**

Publisher: ACM Press

Full text available: pdf(185.74 KB) Additional Information: full citation, abstract, references, citings, index terms

Much prior work has shown that the performance enabled by garbage collection (GC) systems is highly dependent upon the behavior of the application as well as on the available resources. That is, no single GC enables the best performance for all programs

and all heap sizes. To address this limitation, we present the design, implementation, and empirical evaluation of a novel Java Virtual Machine (JVM) extension that facilitates dynamic switching between a number of very different and popular g ...

Keywords: Java, annotation, application-specific collection, dynamic selection, hot-swapping, virtual machine

15 New garbage collection algorithms and strategies: Automatic heap sizing: taking real memory into account

Ting Yang, Matthew Hertz, Emery D. Berger, Scott F. Kaplan, J. Eliot B. Moss
October 2004 **Proceedings of the 4th international symposium on Memory management ISMM '04**
Publisher: ACM Press

Full text available: pdf(879.86 KB) Additional Information: full citation, abstract, references, citings, index terms

Heap size has a huge impact on the performance of garbage collected applications. A heap that barely meets the application's needs causes excessive GC overhead, while a heap that exceeds physical memory induces paging. Choosing the best heap size a priori is impossible in multiprogrammed environments, where physical memory allocations to processes change constantly. We present an automatic heap-sizing algorithm applicable to different garbage collectors with only modest changes ...

Keywords: garbage collection, paging, virtual memory

16 Atomic incremental garbage collection and recovery for a large stable heap

Elliot K. Kolodner, William E. Weihl
June 1993 **ACM SIGMOD Record , Proceedings of the 1993 ACM SIGMOD international conference on Management of data SIGMOD '93**, Volume 22 Issue 2
Publisher: ACM Press

Full text available: pdf(1.34 MB) Additional Information: full citation, abstract, references, citings, index terms

A stable heap is storage that is managed automatically using garbage collection, manipulated using atomic transactions, and accessed using a uniform storage model. These features enhance reliability and simplify programming by preventing errors due to explicit deallocation, by masking failures and concurrency using transactions, and by eliminating the distinction between accessing temporary storage and permanent storage. Stable heap management is useful for programming lang ...

17 A practical parallel garbage collection algorithm and its implementation

Yasushi Hibino
May 1980 **Proceedings of the 7th annual symposium on Computer Architecture ISCA '80**
Publisher: ACM Press

Full text available: pdf(560.94 KB) Additional Information: full citation, abstract, references, citings, index terms

One of the major problems of list processing programs is that of garbage collection. This paper presents a new practical parallel garbage collection algorithm and its improvements, and proposes a special processor for parallel garbage collection. For the parallel garbage collection system, an urgent requirement is to reduce the the garbage collector cycle time that is defined as the total execution time for the marking and reclaiming phase. The effect of improvements discussed he ...

18 Generational garbage collection for Haskell

Patrick M. Sansom, Simon L. Peyton Jones

July 1993 **Proceedings of the conference on Functional programming languages and computer architecture FPCA '93**

Publisher: ACM Press

Full text available: pdf(1.20 MB) Additional Information: full citation, references, citings, index terms

19 Controlling fragmentation and space consumption in the metronome, a real-time garbage collector for Java

David F. Bacon, Perry Cheng, V. T. Rajan

June 2003. **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems LCTES '03**, Volume 38 Issue 7

Publisher: ACM Press

Full text available: pdf(354.15 KB) Additional Information: full citation, abstract, references, citings, index terms

Now that the use of garbage collection in languages like Java is becoming widely accepted due to the safety and software engineering benefits it provides, there is significant interest in applying garbage collection to hard real-time systems. Past approaches have generally suffered from one of two major flaws: either they were not provably real-time, or they imposed large space overheads to meet the real-time bounds.Our previous work [3] presented the Metronome, a mostly non-copying real-time co ...

Keywords: compaction, cost model, fragmentation, space bounds

20 New garbage collection algorithms and strategies: Garbage-first garbage collection

David Detlefs, Christine Flood, Steve Heller, Tony Printezis

October 2004 **Proceedings of the 4th international symposium on Memory management ISMM '04**

Publisher: ACM Press

Full text available: pdf(199.59 KB) Additional Information: full citation, abstract, references, citings, index terms

Garbage-First is a server-style garbage collector, targeted for multi-processors with large memories, that meets a soft real-time goal with high probability, while achieving high throughput. Whole-heap operations, such as global marking, are performed concurrently with mutation, to prevent interruptions proportional to heap or live-data size. Concurrent marking both provides collection "completeness" and identifies regions ripe for reclamation via compacting evacuation. This ev ...

Keywords: concurrent garbrage collection, garbage collection, garbage-first garbage collection, parallel garbage collection, soft real-time garbage collection